Inventory and Warehouse Management System

Introduction

Efficient inventory and warehouse management is essential for smooth supply chain operations, ensuring products are available at the right place and time. Businesses today rely on integrated systems to maintain accuracy and minimize losses. This project focuses on designing and implementing a SQL-based backend system for managing products, suppliers, warehouses, and stock levels. The system enables accurate tracking, automated low-stock alerts, and stock transfer functionality between warehouses, which is critical for operational efficiency.

Abstract

The Inventory and Warehouse Management System was built using MySQL as the backend database platform. It includes schema design for suppliers, products, warehouses, and stock tables. The design supports realistic inventory operations by inserting large datasets such as 50+ suppliers, 50+ warehouses, and 50+ products. The system ensures timely low-stock alerts using triggers and supports stock transfers via stored procedures. Queries for stock checks, reorder alerts, and transfer tracking were implemented to test business scenarios. This project highlights database normalization, use of constraints, and modularization of SQL logic for better reusability and reliability.

Tools Used

- **MySQL**: Used as the backend relational database management system.
- **DBeaver / MySQL Workbench**: Used as database management and visualization tools.
- **SQL**: Core language for schema creation, data insertion, queries, triggers, and procedures.
- **Python (ReportLab)**: For generating project reports in PDF format.

Steps Involved in Building the Project

1. Designed the schema with normalized tables for Suppliers, Warehouses, Products, and Stock.
2. Inserted realistic sample data, ensuring adequate volume for testing (50+ suppliers, 50+ warehouses, and 50+ products).
3. Developed SQL queries to check stock levels, generate reorder alerts, and track available inventory.
4. Implemented triggers that automatically activate low-stock alerts whenever inventory falls below defined thresholds.
5. Created a stored procedure that manages inter-warehouse stock transfers while updating logs and validating stock levels.
6. Tested various scenarios such as low stock, threshold overrides, and transfers to ensure the system worked as intended.
7. Documented schema, sample queries, triggers, and procedures for reference and reusability.
8. Generated a structured PDF report to summarize the project outcomes.

Conclusion

The project successfully demonstrates the design and implementation of a SQL-based backend system for inventory and warehouse management. It automates stock monitoring, ensures timely low-stock alerts, and supports stock transfers across multiple warehouses. By simulating realistic business data, the project proves the scalability and adaptability of the system. This system can serve as a foundation for integrating advanced modules such as sales management, purchase tracking, forecasting, and reporting dashboards. Overall, the project provides a strong practical understanding of SQL database design and backend implementation for real-world inventory management.